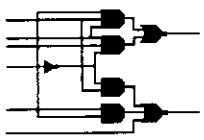
DesignWorks/Converter

File Format Conversion

for Pre-3.0 Files

for the Apple Macintosh™

September 25, 1992



Capilano Computing

406 - 960 Quayside Drive

New Westminster, B.C., Canada, V3M 6G2

(604) 522-6200

©1992 Capilano Computing Systems Ltd.

Table of Contents

Introduction **1**

Major Conversion Issues 1

The Converter Tool **3**

What Converter Does 3
Version Compatibility 4
Converter Installation 4
Converter Operation 5
Converting a Single File 5
Conversion Options 6

The ConvLib Tool **7**

ConvLib Installation 7
ConvLib Operation 8
Library Conversion Issues 10

Conversion Issues **11**

Hierarchy 11
Text Rotation 11
Attributes 12
Sheet Templates 14
Bus Breakouts 14
Packaging 15
Page References 15

Report Issues **16**

Attribute Field Changes 16
Report Form Language Changes 16

Introduction

IMPORTANT: If you are importing circuit files or libraries created by a pre-3.0 version of DesignWorks, please read at least the first two pages of this manual!!! The Converter and ConvLib tools make it easy to import the old files, but there are issues you should know about before proceeding with a major design!

Major Conversion Issues

DesignWorks™ 3.0 includes a number of major new design features and symbol libraries, and is generally much more structured than its predecessors. For this reason, there are some issues that you should be aware of before proceeding to work with a design imported from an older version.

The Big Decision

If you are currently working with designs in a pre-3.0 version of DesignWorks, you can choose one of the following conversion strategies.

Continue Using the Old Version

If you are almost finished with a design, you may decide to continue using the older version for your existing designs until they are completed. This may be the preferred approach since the benefit of using the new version at this stage would be minimal. The 3.0 version has a completely new file type, so there should be no confusion in keeping both versions on your disk.

In any case, when your design is complete, we would recommend converting the files to 3.0 format and saving them for archival purposes. Capilano Computing will no longer be supporting the older versions and it is possible that system hardware or software changes might render them unusable.

Minimal Conversion to 3.0

The recommended strategy is to convert your circuit files to 3.0, but continue using only the DesignWorks features similar to the old version, and continue using the symbol libraries imported from the old version. The

conversion effort required is minimal and you avoid the confusion of using two versions.

In this case, you should do at least the following:

1) Review attribute fields used in your design. The Converter module automatically converts some commonly-used field names to new names. See the section on Attributes later in this manual.

2) If you have created any custom report formats, there are some minor changes that may be needed for them to work under 3.0. We recommend testing your netlist output in the new version before proceeding. See the section on Report issues later in this manual.

Full Conversion to 3.0

Converting an existing design to fully take advantage of new features like gate packaging can be a significant task, involving the following steps:

1) Review attribute field usage and report formats, as noted above.

2) EITHER:

- Enter the packaging attributes into your existing libraries then update the design from the libraries, OR,
- Enter the packaging attributes into the individual devices on the schematic using the Browser, OR,
- Update the schematic to the new libraries. Because of the change to the ANSI standard symbol format, this is normally impractical without completely redrawing the schematic.

The Converter Tool

The Converter tool is a module which converts files created by pre-3.0 versions of DesignWorks (and its predecessors) to the new format. The converted file can then be used directly in the new version.

NOTE: Although the Converter tool does 99% of the work for you, there are some conversion issues that cannot be addressed automatically. These relate especially to attribute field usage. Please see the section below on attributes before proceeding with editing a converted design.

See the section below on the ConvLib tool for converting symbol library files.

The DesignWorks 3.0 release uses a completely new data file format in order to:

- allow for hierarchical design structure and other new features in 3.0
- be compatible across multiple computer types
- be more easily expanded to incorporate future features
- be more reliable and more easily repaired in case of file damage

Because the new file structure is so different from the old one, we felt that there would be a significant penalty in memory usage if the new Schematic module included the ability to read old files. Therefore, the conversion capability has been incorporated into two separate modules. Once all your old files have been converted, you can remove the Converter module to a backup location in order to reduce memory overhead.

What Converter Does

The Converter module performs the following sequence of operations on each circuit file:

- Read file into memory.
- Check file integrity and repair where possible.
- Optionally convert macro device internal circuits to new hierarchical format, or remove them.
- Scan circuit for attribute definitions and create a definition table.

- Break displayed attribute field values into separate items.
- Convert bus breakouts to new format.
- Save file in new format.

Version Compatibility

The Converter module can read any file created by all previous released versions of Capilano schematic entry products, including:

- LogiMac 1.0 - 1.3
- LogicWorks 1.0 - 2.5
- DesignWorks 1.0 - 2.5.6

Converter Installation

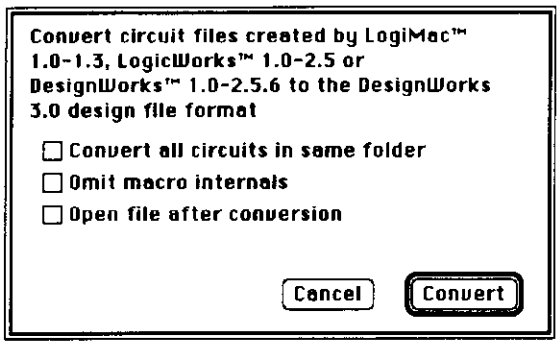
The Converter tool is implemented as a MEDA code module. This means that it must be either in the same folder as the DesignWorks program, or in the Tools folder (or other tool folder specified in the DesignWorks Setup file).

Memory Usage

The Converter tool reads the entire old design into memory and performs a number of operations on it. It is recommended that the Finder memory size allocation for DesignWorks be set so that the available memory (as shown on the Schematic tool palette) is at least twice the size of the largest file you will be converting.

Converter Operation

The Converter module is invoked by selecting its name in the Tools menu. It will display the following box:


Convert circuit files created by LogiMac™ 1.0-1.3, LogicWorks™ 1.0-2.5 or DesignWorks™ 1.0-2.5.6 to the DesignWorks 3.0 design file format
Convert all circuits in same folder
Omit macro internals
Open file after conversion
Cancel
Convert

Converting a Single File

To perform a file conversion, follow these steps:

- Click on the "Convert" button. A standard file selection box will allow you to select the file to convert.
- A save file box will then allow you to select the location and name for the output file.
- A box will announce progress of the conversion and, unless an error occurs, the new file will be saved as specified.

IMPORTANT: If the option "Convert all files in same folder" is checked, the name entered in the save file box is scanned to find a file suffix (i.e. a period "." followed by some characters). If none is found, the default ".DW3" is used. This suffix will be appended to each input file name to create the output file name.

Conversion Options

Converting All Files in a Folder

The check box "Convert all files in same folder" allows you to convert a batch of files without having to select them individually. To do this:

- Click on the "Convert" button and choose any file in the source folder.
- Enter the desired output file name, preferably the same name plus a "." plus some suffix. This suffix will be appended to each input file name to create the corresponding output file name. The destination folder can be the same as or different from the source folder.
- All files in the specified source folder will now be converted.

Eliminating Internal Circuits

The check box "Omit macro internals" instructs the program to remove any internal circuits from macro devices before converting. This can significantly reduce the size of a circuit file, but will eliminate all simulation functionality from macro devices in the file. If you do not anticipate wanting to simulate the design, then use this option.

Open the File When Done

The Converter module can optionally open the converted file in Schematic when the conversion is complete. This is not recommended when converting more than one large file or a large number of smaller files.

The ConvLib Tool

DesignWorks 3.0 uses a new library file format which addresses the following issues:

- Files can now contain thousands of symbols with major performance degradation.
- Symbol files can be shared across a network.
- Files are opened only when actually being accessed, eliminating problems with open file limitations in the Macintosh system.

The ConvLib tool converts symbol library files used by older versions of DesignWorks to the new format.

ConvLib Installation

The ConvLib tool is implemented as a MEDA module, meaning that it must be either in the same folder as the DesignWorks program, or in the Tools folder, when the program starts up.

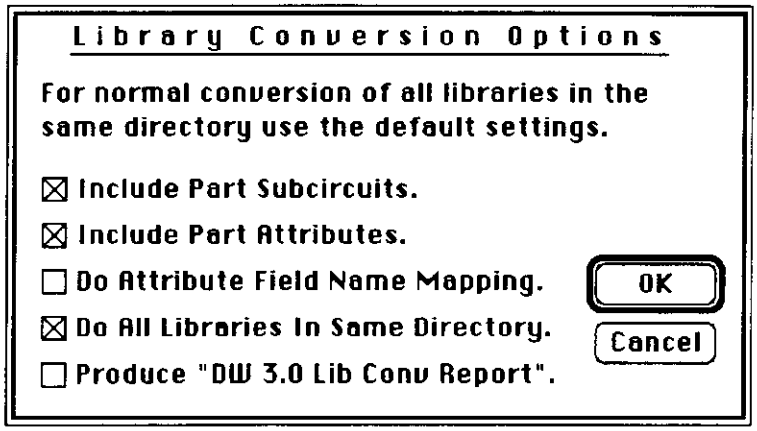
> NOTE: ConvLib works in conjunction with the Converter tool to convert internal circuits of macro devices. ConvLib will not work if Converter is not present.

Memory Usage

ConvLib does not require a large amount of memory unless the library parts contain large internal circuits. A good rule of thumb is to ensure that the available memory display in Schematic is showing at least the size of the largest library file to be converted.

ConvLib Operation

The ConvLib tool is started by selecting its name in the Tools menu. The following box will appear:


Library Conversion Options
For normal conversion of all libraries in the same directory use the default settings.
Include Part Subcircuits.
Include Part Attributes.
Do Attribute Field Name Mapping.
Do All Libraries In Same Directory.
Produce "DW 3.0 Lib Conv Report".
OK
Cancel

To perform a default library conversion, simply click the OK button. You will be presented with a standard open file box. Select the library to be converted, or, if converting all libraries in a folder, select any library in the target folder. The converted libraries will be written in the same folder, with the extension ".DW3" appended to their original name.

ConvLib Options

Include Part Subcircuits

When enabled, this option converts any subcircuits associated with the parts to the new format. If this is disabled, internal circuits will be discarded. If you have no interest in the simulation functions of parts, you may wish to disable this option to reduce the overhead of unneeded internal circuits.

Include Part Attributes

When enabled, this option causes all default attribute values stored with the part to be converted to the new format. If disabled, all attributes in the original parts are discarded.

Do Attribute Field Name Mapping

When you click on this option, you will be prompted for the name of a mapping file. This file contains the old and new names of attribute fields in the following format:

```
oldName1 newName1
oldName2 newName2
         .
         .
         .
```

To completely remove an attribute field, "newName" can be specified as "$Delete" (case is significant). For example:

```
74 $Delete
74LS Part
info Value
```

This file converts the field named "74LS" to "Part", the field named "info" to "Value" and deletes the field named "74".

This file must consist of text only (i.e. saved with TeachText, a programming text editor, or a word processor with a "text only" option. Exactly one pair of names must appear on each line, separated by one or more spaces.

See the section below on attribute conversion for recommended field name mappings.

Do All Libraries in Same Directory

When enabled, this option causes ConvLib to convert all libraries in the same folder as the selected item.

Produce Conversion Report

When you select this option, you will be prompted for the name of a report file. This file lists conversion activity and possible errors.

Library Conversion Issues

Attribute Name Mapping

We strongly recommend reviewing the notes on the new attribute structure in the following section. The Attribute Field Name Mapping feature of ConvLib can be used to convert old usage, e.g. "pkg" to the new standard "Package". See the sample name mapping file provided with the conversion modules.

TYPEFAMILY No Longer Supported

The TYPEFAMILY feature of version 2.5 (which would select an attribute field to be used as the source for the part name) is no longer supported in 3.0. If you wish to convert the old standard libraries to 3.0, or if you have used this feature in your own libraries, we recommend using the name mapping feature to convert your most commonly-used family name to the new standard part name field "Part". See the sample name mapping file provided with the package.

Conversion Issues

Hierarchy

DesignWorks 3.0 implements a true hierarchical design structure. This means that "device" symbols appearing in a circuit can actually represent another, nested circuit. This sub-circuit can be opened and edited at any time in separate circuit window.

Macro Internal Circuits

Older versions of DesignWorks allowed macro devices to contain an internal circuit, but the internals could not be edited once placed in a circuit. For this reason, these files did not store all graphical information for the internal circuits. Once a file is converted, you will be able to open the internal circuits, but they will not contain any signal lines and may have other graphical problems. It is possible to recover an internal circuit by converting the original source library to the 3.0 format and then using the Update from Lib command to update the internal circuit.

Hierarchy Mode

Converted files default to "flat" hierarchy mode since this most closely matches the operation of older versions. Once the converted design is read in by the Schematic module, it is possible to change the mode in the Design Preferences box. Be sure to read the section on hierarchy mode in the Schematic manual before doing this. Switching to a full hierarchy mode has a number of implications for usage of attribute fields, naming conventions, packaging, etc.

Text Rotation

In DesignWorks 3.0, text can now be rotated in 90° increments. Different rules apply for conversion of different types of text:

- Attributes that were displayed on the diagram will appear in the normal (unrotated) orientation to match the original schematic. If desired, attribute text can be rotated using the attribute pop-up menu in Schematic (i.e. holding the Command key and clicking on the attribute text).

• In DesignWorks 3.0, pin numbers on devices are rotated to match the device they are on. In older versions they were always horizontal. If you prefer to revert to the style of your original schematic, you can turn off pin number rotation in the Design Preferences box in Schematic.

• Text inside device symbols is now always rotated to match the symbol graphics. There is no way of disabling this.

• Miscellaneous text items cannot be rotated and will appear in the same orientation as before.

Attributes

The text attributes facility in DesignWorks has been made much more structured and much more powerful in version 3.0. Each design now has associated with it an "attribute definition table" which lists all the fields which will be used in the design.

The Converter module scans the entire incoming circuit for attributes and builds a definition table. When the design is opened in the Schematic module, all these fields will now appear in the design's attribute definition table.

Attribute Conversion Restrictions

Field Name Restrictions

In DesignWorks 3.0, attribute field names are limited to 15 characters, and must consist of letters, numbers and the underscore "_" character. Case is significant.

In the conversion process, the following name conversion is done:

- The character "+" is converted to the word "Plus".
- The character "-" is converted to the word "Minus".
- All other non-alphanumeric characters are converted to an underscore.
- The name is truncated to 15 characters.

Data Format Restrictions

The Converter and ConvLib modules can only convert attribute data that was formatted in the standard way. Data without a recognizable field name followed by an "=" followed by a value will not be converted.

Reviewing the Converted Attributes

For any design that is going to be edited extensively in version 3.0, it is worth reviewing the usage of attribute fields and checking for the following:

- Where possible, use the standard, pre-defined fields for common functions. For example, if you were using a field called "Pkg" for the device package type, convert this to the standard field "Package". This can be done by simply renaming the old field to the new one in the Define Fields box in the Schematic module. You will be asked if you wish to merge the old field into the new, to which you should respond with the Merge button. NOTE: The Converter tool performs some conversions of commonly-used names automatically. These are summarized in the next section.

- Since the field name was entered separately into each object in the older version, it was easy to create an inconsistent set of field names with different spellings or upper/lower case usage. For example "pkg" and "Pkg" might have been used for a package type. When you read the design into version 3.0, use the Define Attribute Fields menu command to check for these inconsistencies. You can use the Delete and rename features in the Define Fields box to merge or remove inconsistent data.

Automatic Field Name Conversions

The Converter tool automatically converts some commonly-used attribute field names to the new standard. These are summarized in the following table.

Old Name	New Name	Comments
+5V	Power	Main power pins
+XXX	PlusXXX	Other positive power pins
-XXX	MinusXXX	Other negative power pins
Gnd	Ground	Ground pins
PAGECONN	PageRef	Page connector annotations
pkg	Package	Package type code
value	Value	Component value or information

Converted fields are always marked as "primary" so they will appear in the Attribute Probe box, in the Browser's "Primary" list, and in the "Primary Fields" list for all objects.

Sheet Templates

DesignWorks 3.0 allows all aspects of sheet sizes, borders, grid layouts, title blocks and text annotations to be customized. The package now has no built-in knowledge of standard sheet sizes, but instead imports all information from "template files", which are actually just empty design files. Text and graphics can be placed on a sheet and marked as "border" items. They will then be automatically copied to each new page in the design and can be imported to other designs with a single command.

The Converter module automatically converts the border format so that it will appear identically in the new version.

Bus Breakouts

Signal Names

Bus breakouts are implemented somewhat differently in DesignWorks 3.0. Instead of the signal names being part of the symbol for the breakout "device", they are now stored as pin attributes. This allows the names to be edited at any time after the breakout is created.

Pin Length

In order to allow space for this attribute text, the signal pins on the breakout are lengthened by the width of the longest name. This length cannot be adjusted after the breakout is created. For this reason it is still best to delete and recreate the breakout if much longer or shorter names are needed.

Spine Length

The breakout "spine" (the long bus pin perpendicular to the signal pins) is normally created shorter in DesignWorks 3.0. Converted circuits will have the same spine length as the original circuit. If you wish to take advantage of the new look, it will be necessary to delete and replace the breakouts in your circuit manually.

Packaging

DesignWorks 3.0 now has an "auto-packaging" feature which automatically assigns names and gate unit numbers to devices. This feature relies heavily on special attribute fields present in the new DesignWorks 3.0 libraries. These fields indicate how many gate sections exist in a package and the corresponding pin assignments.

Since this data will not present in converted designs, the Packager module will assume that every device in the circuit is completely distinct and will not assign multiple gates to a single package. In order to use the packaging feature it will be necessary to either replace all gate devices (i.e. all devices with multiple units per package) with new library parts, or enter the required attributes manually. This is not as painful as it sounds since the Browser allows easy entry of data into multiple devices, but is still an exacting operation.

See the section on Packaging in the Schematic manual for details on the attribute fields used for this purpose.

Page References

The automatic page references that were created in older versions of DesignWorks will be transferred verbatim to the converted design. However, the new version allows you a number of options which you can take advantage of:

- auto-page reference updating can be turned off completely to eliminate the processing delays encountered with larger designs. Page references can then be updated manually when desired.
- there is considerable flexibility in the format of the page references, including the "Page-Grid" format and the maximum line length.

These options are all controlled from the Design Preferences command in the Schematic module.

Report Issues

Attribute Field Changes

If you have created any custom report formats for use with older versions of DesignWorks, they should be updated to reflect the new standard attribute field names. See the section earlier in this manual on attribute conversion.

All standard report forms supplied with the package have been updated to reflect the new usage.

Report Form Language Changes

All existing report forms can be used with DesignWorks 3.0, but some subtle changes to the language may affect the results. These are summarized in the following sections.

Find and Sort

The functions of the $FIND and $SORT commands have been changed somewhat in version 3.0. Simple, single-line sorts or finds will still work as before. More complex operations will probably have to be reformatted as follows:

• Each $FIND is now completely independent and undoes all previous $FINDs or $SORTs. The $CLEAR keyword is no longer required or implemented. If you have multiple $FIND lines in your format that rely on the fact that each $FIND was a subset of the previous one, they will have to be merged into a single line.

• Each $SORT is now completely independent and undoes all previous $SORTs. (Note that $SORT does not clear the last $FIND, even though $FIND does clear the last $SORT.) The $CLEAR keyword is no longer required or implemented. If you have multiple $SORT lines in your format that rely on the fact that each $SORT was a subsort of the previous one, they will have to be merged into a single line.

• To clear a SFIND (i.e. revert to the complete list of objects), simply do a find with no criteria, either:

```
$FIND $DEVICES
```

or

```
$FIND $SIGNALS
```

Null Signal Selection

In 3.0, signal reports will by default include all signals in the circuit, including null signals (i.e. with 1 or less connections). The old keywords $NULLSIGSON and $NULLSIGSOFF are still supported but not recommended. A more explicit way of filtering these signals out is:

```
$FIND $SIGNALS $NUMPINS(2)
```

In either case, the internal operation done is the same.

Other Changes

A number of keywords have been updated to make the language more consistent and support new features. Where possible, the old keyword is still supported for compatibility, although it will not appear in the Report chapter in the Schematic manual. Some changes include (* indicates format changes required other than just the keyword change):

Old Keyword	New Keyword
$CIRCUITNAME	$DESIGNNAME
$CREATOR (on separate line)	$CREATEREPORT *
$DEVNUM	$DEVSEQ
$FILE	$DESIGNNAME
$PINFORMAT	$SIGPINFORMAT and $DEVPINFORMAT
$SIGNALS (on $DEVICES line)	$PINS
$SIGNUM	$SIGSEQ
$SUFFIX	$CREATEREPORT *

No Longer Supported

The following keywords are not supported in this version:

> NOTE: Simulation-related items are not supported in the initial release of DesignWorks 3.0. They may be implemented in later upgrades.

```
$ATTRIBUTES
$DELAY
$EVENTS
$LINES
$PAGEINFO
$SIGTIME
$SIGVALUE
$TYPECOUNT
```

SearchLib

DesignWorks™ Library Browser

Version 1.0

for the Apple Macintosh™

Reference Manual

Revised: September 23, 1992

406- 960 Quayside Drive

New Westminster, B.C., Canada, V3M 6G2

(604) 522-6200

©1992 Capilano Computing Systems Ltd.

Table of Contents

Introduction 1

Installation 1
Memory Usage 1
Starting SearchLib 1

Using SearchLib 2

Finding the MainIndex 2
Searching for Library Components 3
Placing Symbols from the SearchLib Dialog 5
Closing the SearchLib Dialog 6

Error Messages 7

The library folder is not in the application folder. 7
Please Locate the "MainIndex" file 7
Error Opening Index File: 7
Error in Search Options: 7
An error occurred while opening library: xxx 8
Too many parts found. 8
Out of memory 8

Introduction

SearchLib is a MEDA module used to search for parts in the DesignWorks Libraries. It allows you to quickly locate library parts given different criteria such as category, function or part name.

NOTE: The SearchLib tool has been included in the initial DesignWorks 3.0 release as a product under development. This tool was developed late in the testing stages of DesignWorks 3.0 to ease the task of finding parts in the new libraries. The SearchLib tool will be undergoing continuing development and will be formally released in a future update. Any comments you may have on it would be appreciated.

Installation

SearchLib is a tool that works with DesignWorks using the MEDA interface. To locate the tool when DesignWorks starts up, it must be in one of two places, the same folder as the DesignWorks program, or in the folder specified in the TOOLFOLDER statement in the DesignWorks setup file (this is "Tools" by default).

Memory Usage

The SearchLib module occupies a small amount of memory space, about 82K. It may however require a much larger amount of memory depending on the amount of information found during a search. The SearchLib module builds a structure in memory for each item found during a search.

Starting SearchLib

To invoke SearchLib select it from the Tools menu. If all your SearchLib Index files and Library folders are in the normal place the SearchLib window should appear. If you have moved any of the files that SearchLib uses then you will be prompted to locate where they have been moved to.

Using SearchLib

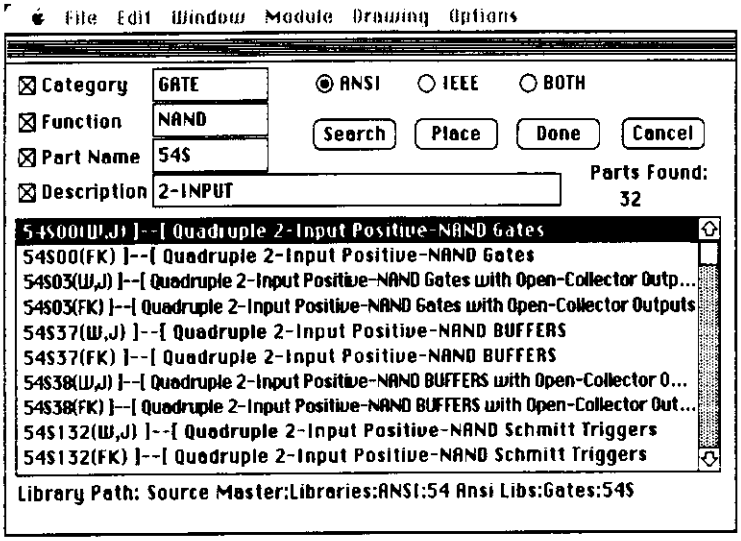
The SearchLib Tool allows the contents of the libraries to be browsed. It allows searching by several criteria and displays a text description of each component that was found. SearchLib also allows any component displayed in the found list to be selected and placed in a schematic. This saves much manual searching through the 12000+ components when you do not quite know which part you want or alternatively you do not know which library it is stored in (or if it exists in the libraries at all).

Finding the MainIndex

When the SearchLib Tool is first selected after launching DesignWorks it may show a message asking you to locate the MainIndex file. This dialog will be followed by a file selection dialog. There are four text files that are used by the SearchLib tool, these are "MainIndex", "DirIndex", "CatIndex" and "FuncIndex". These files should be in a folder within your DesignWorks folder. Use the file selection dialog to open the "MainIndex" file. This prompt will only appear if the index files are not in the default location. By default the module looks for ":SearchLib:SearchLib Index Files:MainIndex". After selecting the Index, SearchLib looks for a folder called "Libs". If the folder is not found then the module will prompt you to find the "LibRoot" file in the library folder.

NOTE: The SearchLib tool uses a set of index files that contain the directory structure, Library names, Part names and descriptions for each part in the libraries. If you alter the structure of the "Libs" folder, rename a library or rename components in a library SearchLib will not be able to find those components for placement. In this case you will get an error from the LibIO module when you try to place a part from the SearchLib module.

Searching for Library Components



The SearchLib tool allows the libraries to be searched by five different criteria:

- Component Category.
- Component Function.
- Library Part Name.
- Text description for the component.
- Symbol type, IEEE, ANSI or BOTH.

The **Part Name** text is matched against the library name. This is the name that appears in the parts palette. The name will match any name containing the given string. Thus "68" will match "6809" as well as "54F168".

The **Description** text is matched against the contents of the Description attribute for each part. It is matched against any text containing the given string. So "2-input" will match "Quad 2-Input NAND". The text that appears in the scrolling list contains the description so it can be used as a template when selecting suitable search strings.

The **Category** and **Function** selections are text strings chosen from the following table. Each row contains the Category followed by all the valid Functions that may be associated with it:

Category	Functions
ARITH	ACCUM, CARRY, ALU, MULT, ADD
BUFF	BUFFINV, DELAY, BUFF, INV
COMP	ADDCOMP, ERRDET, PARITY, COMP
CONN	CONNM, CONNF
CTR	PRESCALE, TIMER, OSC, FREQDIV, RATEMULT, CTR
DIODE	TUNNEL, LED, BRIDGE, ZENER, DIODE
ELMECH	RELAY, SWITCH
FF	DFF, JKFF
GATE	OAI, XNOR, GATE, XOR, AOI, OR, AND, NOR, NAND
LATCH	MULTIVIB, LATCH
LIN	ADCONV, VOLTREG, LIN, OPAMP
MEM	ROM, EPROM, EEPROM, PROM, DRAM, FIFO, SRAM
MISC	LVLCNV, PLL, RCVR, MISC, FILTER, DSWITCH, PLSSYNCH
MUX	AMUX, BITSHIFT, MUX, ENCODE, DECODE
NETSYM	PWR, GND
PASS	XTAL, XFMR, TEST, RES, IND, FUSE, CAP, BAT
PLD	PLD, EPLD, PAL, FPLA
PROC	COPROC, MPERIPH, MCTRL, PROC, MEMCTRL, MEMMAP
REG	REG, SHREG
THYR	UJT, TRIAC, THYR, SCR, DIAC
TRANS	TMOS, PNP, NPN, JFET, MOSFET
XCVR	XCVR

There are three **radio buttons** that allow you to select the symbol style you wish to search for:

- **ANSI** will only find symbols of the ANSI style.
- **IEEE** will only find symbols of the IEEE style.
- **BOTH** will find ANSI and IEEE symbols.

If you have specified BOTH you can tell which components in the list of found items are ANSI or IEEE by looking at the Library path at the bottom

of the dialog, or at the suffix of the file name. Possible suffixes include A for ANSI, I for IEEE, AM for ANSI monolith, IM for IEEE monolith and D for DeMorgan. If the path name shows IEEE in it then the component is an IEEE symbol and if the path shows ANSI then it is an ANSI symbol.

The four buttons in the dialog allow you to search for components using the given Category, Function, PartName, Description and Symbol type, Select a component for placement in a schematic and close the SearchLib dialog with or without saving the current list of found components.

The **Search** button can be selected by either clicking on it or pressing the Return key. When search is selected any of the four text boxes whose check box is checked will be used to match against library items. Searching the index files for components can take several minutes on a slower Macintosh. When the search is completed the number of matches found will be displayed under the Parts Found in the dialog. If there were more than 100 matches then a dialog indicating that only 100 of the matches will be displayed.

If there were no matches the module issues a short beep and the parts found will display 0.

After a successful search a list will be displayed in a scrolling window showing the library part name followed by a separator "]--[" then a text description of what the part is.

If you click once on one of the items in the list the full pathname and library name for that item will be displayed at the bottom of the dialog next to the words "Library Path:".

Placing Symbols from the SearchLib Dialog

The **Place** button will attempt to open the library of the selected (hi lighted) item in the list produced by the search. The dialog will close and the indicated component will appear on the cursor for placement in the schematic. An open schematic window must be present for the component to place. Double clicking on an item in the list is equivalent to pressing the Place button. If there are no selected items in the list then the place button has no effect. If the SearchLib module is selected again the same list will still be available for placing further components.

Closing the SearchLib Dialog

The **Done** button will close the SearchLib module and free all the list items. This means that if you subsequently select the SearchLib module the list will be empty.

The **Cancel** button will close the dialog without freeing the list items. This means that if you bring up the dialog again the items in the list will still be available for selecting and placing.

Error Messages

The library folder is not in the application folder.

This error is produced when setting the path to the "Libs" folder. The default location for the "Libs" folder is in the "DesignWorks 3.0" folder. The tool will prompt you to locate the "LibRoot" file which should be in the "Libs" folder. The error probably means that you moved your libraries folder from where it was initially installed or it was renamed to something other than "Libs".

Please Locate the "MainIndex" file...

This error is issued when the SearchLib tool is invoked from the Tools menu when any of the SearchLib index files could not be found or opened. The files "DirIndex", "LibIndex", "CatIndex", "FuncIndex" and "MainIndex" must all be present in the same folder. The tool will prompt you to locate the "MainIndex" file. This error probably means that the "SearchLib" folder was moved or renamed.

Error Opening Index File:...

This error is issued when any of the SearchLib index files could not be found or opened. The files "DirIndex", "LibIndex", "CatIndex", "FuncIndex" and "MainIndex" must all be present in the same folder.

Error in Search Options:

Invalid Function Name - The Function specified is not one of those in the valid function list (See above).

Invalid Category Name - The Category specified is not one of those in the valid category list (See above).

Valid Functions for Category xxx are "...". - Gives a list of valid Functions that may be used.

Invalid Category for function xxx. Use yyy. - Gives the Category that may be used with the given function.

An error occurred while opening library: xxx.

The library selected when placing a component could not be found. This can result when the libraries have been moved or renamed.

Too many parts found.

There is a maximum of 100 parts that may be displayed in the scrolling list of the SearchLib dialog. When a search results in more than 100 items this error is displayed. It is just a warning to let you know that the list displayed is not complete. The total number of matches is given.

Out of memory.

The SearchLib module has run out of memory. This error occurs when there is less than 64K bytes of total available memory to the application. If you have a large design and you have been doing a lot of searching with the SearchLib module it is possible that memory will become fragmented. To resolve this problem you can quit DesignWorks and restart it.

Guide to Libraries

DesignWorks™ 3.0 Library Reference

Version 1.0

for the Apple Macintosh™

Reference Manual

Revised: September 23, 1992

406- 960 Quayside Drive

New Westminster, B.C., Canada, V3M 6G2

(604) 522-6200

©1992 Capilano Computing Systems Ltd.

Table of Contents

Introduction 1

About the Libraries 2

Symbol Format 2
Finding a Library 3
Interpreting Library Part Names 4
Selecting the Unit and Part 6

Attributes Defined in the Library Symbols 8

Attributes on types 8
Name.Prefix 8
Name.Spice 8
Name.Pt 8
Unit 8
Unit.List 8
Unit.All 8
UnusedPins 9
Power 9
Ground 9
Restrict 9
Permutable 9
Function 10
Category 10
Description 11
Part 11
Part.List 11
Part.Pt 11
Package 11
Package.List 12
Attributes on Pins 12
VisPin.List 12

The Permutable Attribute 13

Package Codes 16

Introduction

The libraries provided with DesignWorks 3.0 contain over 12,000 symbols including both ANSI and IEEE format symbols. The purpose of this manual is to describe the naming, graphical, attribute and organization conventions used in the libraries. This information will be of interest if you if you need to adhere to some particular drawing standard or if you plan to export data from DesignWorks to another system.

About the Libraries

Symbol Format

The DesignWorks 3.0 libraries include the ANSI and IEEE symbol equivalents for each part. The installation procedure allows you to choose to install only one symbol type.

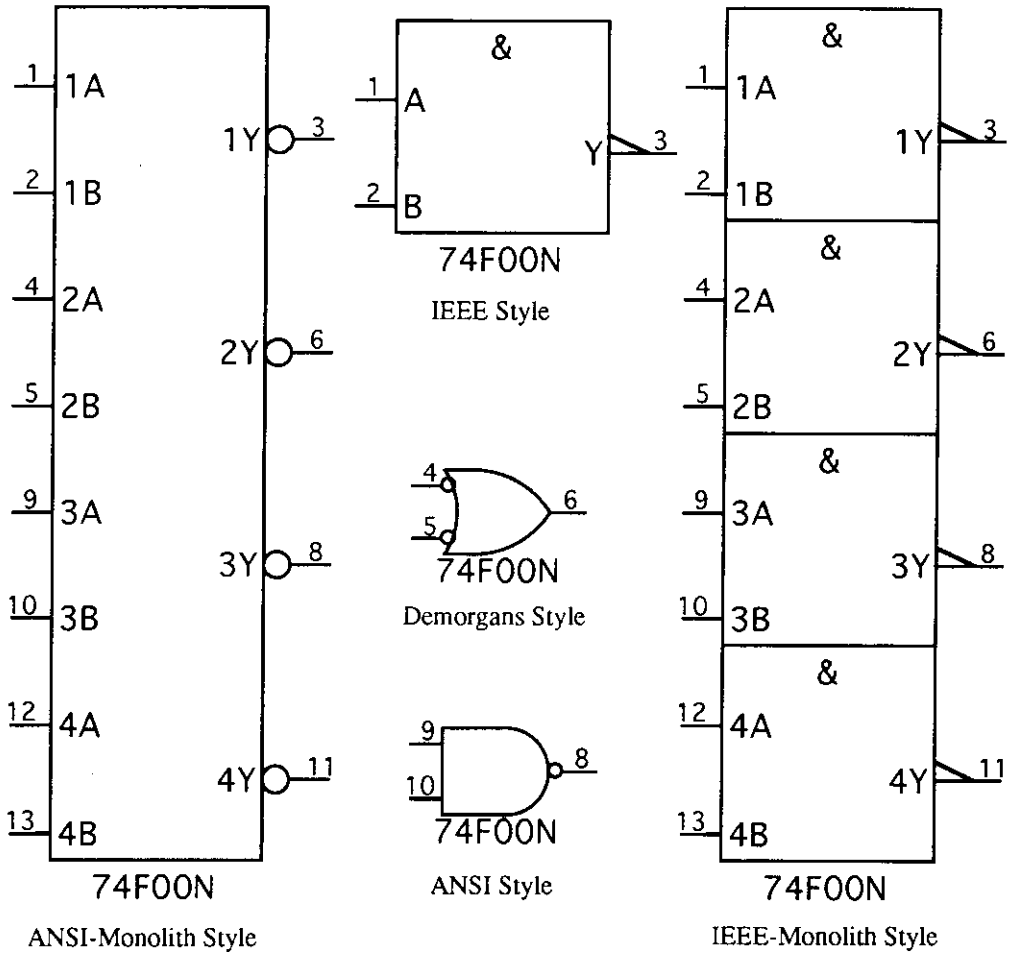


Figure 1 - The various symbol styles available

Finding a Library

See also the SearchLib manual for information on searching for a specific part by characteristics.

The directory structure used to store the libraries is divided into several folders. The top most folder is separated into Analog, ANSI and IEEE.

Within Analog folder you will find:

- Connectors
- Diodes
- Discretes
- National Linear
- Transistors
- Spice Lib
- Misc. Analog

Within the ANSI and IEEE folders you will find:

- 54 TTL
- 74 TTL
- 75 TTL
- CMOS
- ECL
- Memory
- Micro Processors
- Miscellaneous
- PLD

The 54, 74 and 75 series components are divided into their families, so there are separate libraries for the expanded components in the folder "Expanded" and non-expanded components in the "Monolith" folder. Within the Expanded folder where appropriate there is also a "DeMorgans" folder that contains the DeMorgan equivalent symbols for many of the components. All the TTL components are derived from Texas Instruments data.

Within the TTL folders are the following library files:

74, 74S, 74LS, 74AS, 74ALS, 74AC, 74ACT, 74F, 74HC, 74HCT, 74HCU.

54, 54S, 54LS, 54AS, 54ALS, 54AC, 54ACT, 54F, 54HC, 54HCT, 54HCU.

75ALS.

Memories are divided into Motorola, Texas Instruments (TI) and Toshiba.

Micro Processors are divided into Intel, Motorola and Zilog.

Interpreting Library Part Names

A component name is made of up to five parts.

The Logic Family Name

The Device Number

Device Attributes

DesignWorks unit for packaging

A marker indicating the symbol style

Most of these name parts are only used for the TTL family of parts. Other components may have just the manufacturer part name or a descriptive name as the library name, for instance "DIODE" or "2N222".

Some parts, like diodes, have over 100 different names in the parts list. In this case the library part name will be the first name that appears in the Part.List attribute.

Occasionally the part name is too long to display in the parts palette list, in this case the name will appear truncated.

The Logic Family Name

The logic family name is usually from one of the following:

74, 74S, 74LS, 74AS, 74ALS, 74AC, 74ACT, 74F, 74HC, 74HCT, 74HCU.

54, 54S, 54LS, 54AS, 54ALS, 54AC, 54ACT, 54F, 54HC, 54HCT, 54HCU.

75ALS.

The Device Number

The device number is the middle digits in the component name and corresponds to the symbol type. For example "123" in 74ALS123J.

The Device Attributes

The device attributes are the manufacturer attributes that apply to this symbol. The attributes are enclosed in parentheses "()" such as (W,J).

Unit for Packaging

The presence of a unit indicates that there are multiple symbols required to represent all the units in this package. The unit is appended to the name with a "." so ".a" would indicate that this component will default to unit "a".

If there are different symbols for some of the units within a package then the different units will each have their own entry in the library. The unit letter is appended to the end of the library name such as: "5450(J).a" and "5450(J).b". This component is a Dual 2-Wide 2-Input AND-OR-INVERT Gate (One Gate Expandable). In some cases the part that shows up as xxx.a will have a "c" and "d" units and the "b" unit will be a different entry because it does not share the same symbol.

Symbol Style

The symbol style marker is one of the following:

A	This is an ANSI style symbol.
I	This is an IEEE style symbol.
D	This is a DeMorgan equivalent style symbol.
AM	This is an ANSI Monolith (non-expanded gates).
IM	This is an IEEE Monolith (non-expanded gates).

Example 1:

A 54S00J may be listed in the library selection palette as "54S00(W,J)A"

"54S" is the TTL family name, this part of a library name is only used for the TTL components.

"00" is the component number

"(W,J)" indicates that there are two part suffixes for the 54S00 a "54S00J" and a "54S00W".

The "A" suffix indicates that this part is an ANSI style symbol.

Example 2:

A 5450J may be listed in the parts palette as "5450(W,J).aI".

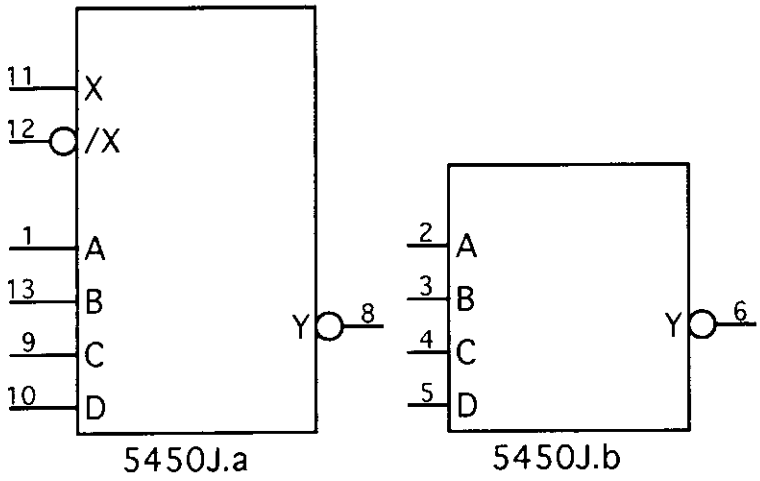
"54" is the TTL family name, this part of a library name is only used for the TTL components.

"50" is the component number

"(W,J)" indicates that there are two part suffixes for the 5450 a "5450J" and a "5450W".

There may be a ".a" (or ".b", ".c" etc...) after the part suffix as in "5450(J).a". This is the unit and it indicates that there is more than one symbol for the gates in a package. Most of the TTL gates have multiple units in a package, the unit name only appears in the library if the graphic symbol for one or more of the units differs from the others.

The "I" suffix indicates that this part is an IEEE style symbol.



Selecting the Unit and Part

After a symbol has been placed on a design page it will have the default unit, part and package values. The defaults are always the first items in the Unit.List, Part.List and Package.List type attributes.

To change the values for the Unit and Part attributes you must use the device Pop-Up menu. This is done by holding the " " key down and pressing the mouse button with the mouse cursor over top of the device.

The Unit attribute is set by selecting a unit from the Unit menu item.

The Part attribute is set by selecting a part from the Part menu item.

The Package attribute is set automatically when a new part from the Parts menu is selected.

If there is no Part.List or only one item in the list then the Parts menu item will not appear in the device Pop-Up menu.

If there is no Unit.List or only one item in the list then the Units menu item will not appear in the device Pop-Up menu.

Example

If you want a 54LS37W, Quad 2-input Positive Nand Buffer, you would open the library named 54LS and select the part "54LS37(W,J)" from the parts palette. To select between the 54LS37W and 54LS37J use the *device pop-up* menu by holding the "⌘" key and pressing the mouse button over the part. In this menu the *part* menu item is used to select which of the valid parts you wish to use.

Also in the device pop-up menu is a unit entry. This is used to select which of the units in a package you wish. For example in a quad nand gate there are 4 units (a,b,c,d). The default is unit a, selecting unit b causes the pin numbers on the symbol to change to the second nand gates pin numbers. The packager tool (described elsewhere) will do this automatically for you when you are placing parts.

Attributes Defined in the Library Symbols

Attributes on types

Name.Prefix

This is the string used as a prefix to auto name a device when it is placed.

Name.Spice

This is the string used as an optional prefix when creating designs for SPICE-based analog simulators.

Name.Pt

This type attribute contains a formatted string specifying the X and Y coordinates of the Name attribute that is generated when a symbol is placed. The format for the string is "(X-coord, Y-coord)". For example "(150, -75)". The values are in 1000ths of an inch and represent the point relative to the symbol. For more details on how to set this see *Default Position Fields* in the manual section on Attributes.

Unit

This is the unit that will be selected as the default unit when a device is placed. It is usually "a".

Unit.List

This type attribute field contains a list of the units available for this symbol. The unit names are separated by commas, e.g. "a,b,c,d". In types having different symbols for the logical units in one package, this is the list of only those units which can be used with this symbol.

Unit.All

This type attribute field contains a list of all the units available in this package, regardless of whether or not they apply to this symbol. Normally

this will be the same as "Unit.List", but will be different for devices having more than one symbol for the logical units in a package, for instance an And-Or-Invert gate.

UnusedPins

This type attribute contains a list of the unused pins in this package. The list is separated by commas, e.g. "13,23".

Power

This type attribute contains a list of the pins connected to power sources in this symbol. The list is separated by commas, e.g. "1, 20".

Ground

This type attribute contains a list of the pins connected to ground in this symbol. The list is separated by commas, e.g. "10, 39".

Restrict

This type attribute is an integer value from 0 to 7 used to set access restrictions for the device's internal circuit. The value is the sum of the following:

1 = Don't report

2 = Don't package

4 = Don't Push Into

The normal setting for this in library parts is 3 (don't report and don't package).

Permutable

This type attribute contains a list of the swappable pins and gates in this package. It is a list of bracketed pin numbers. Pin numbers are separated by commas, Parentheses "()" enclose items that can be swapped while brackets "[]" enclose items that cannot be swapped. This attribute is intended for use by PCB packages. For more information on the format of the Permutable attribute see the description elsewhere in this manual .

Function

This type attribute contains a terse symbol function code. A full list of the currently used function codes and category codes is given below.

Category

This attribute contains a terse symbol category code. A full list of the currently used function codes and category codes is given in the table below. This list will of course be augmented as required for future libraries.

Category	Functions
ARITH	ACCUM, CARRY, ALU, MULT, ADD
BUFF	BUFFINV, DELAY, BUFF, INV
COMP	ADDCOMP, ERRDET, PARITY, COMP
CONN	CONNM, CONNF
CTR	PRESCALE, TIMER, OSC, FREQDIV, RATEMULT, CTR
DIODE	TUNNEL, LED, BRIDGE, ZENER, DIODE
ELMECH	RELAY, SWITCH
FF	DFF, JKFF
GATE	OAI, XNOR, GATE, XOR, AOI, OR, AND, NOR, NAND
LATCH	MULTIVIB, LATCH
LIN	ADCONV, VOLTREG, LIN, OPAMP
MEM	ROM, EPROM, EEPROM, PROM, DRAM, FIFO, SRAM
MISC	LVLCNV, PLL, RCVR, MISC, FILTER, DSWITCH, PLSSYNCH
MUX	AMUX, BITSHIFT, MUX, ENCODE, DECODE
NETSYM	PWR, GND
PASS	XTAL, XFMR, TEST, RES, IND, FUSE, CAP, BAT
PLD	PLD, EPLD, PAL, FPLA
PROC	COPROC, MPERIPH, MCTRL, PROC, MEMCTRL, MEMMAP
REG	REG, SHREG
THYR	UJT, TRIAC, THYR, SCR, DIAC
TRANS	TMOS, PNP, NPN, JFET, MOSFET
XCVR	XCVR

Description

This type attribute contains a short description of the symbol type. For instance "Quadruple 2-Input Positive-NAND Gates".

Part

This type attribute contains the part name that this symbol represents. The value for this attribute is the manufacturer part name, e.g. "74ALS00AN".

Part.List

This type attribute contains a list of all the part names that this symbol may represent. The values for this attribute are the manufacturer part names, e.g. "74ALS00AN, 74ALS00AD". This list is linked to the Package.List, see Package.List below.

Part.Pt

This type attribute contains a formatted string specifying the X and Y coordinates of the Part attribute that is generated when a symbol is placed. The format for the string is "(X-coord, Y-coord)". For example "(150, -75)". The values are in 1000ths of an inch and represent the point relative to the symbol. For more details on how to set this see *Default Position Fields* in the manual section on Attributes.

Package

This type attribute contains the package code for the part defined in the "Part" attribute. For a description of currently used package codes see the description elsewhere in this manual.

Package.List

The list of package codes that are used by this symbol type. The list is separated by commas where each element corresponds to the package used for the part that holds the same position in the Part.List string.

For example:

if the Part.List = "74ALS00AN,74ALS00AD".

then the Package.List = "TI14DPN,TI14SN",

where TI14DPN is the package for the first element of the Part.List 74ALS00AN and so on.

Attributes on Pins

VisPin.List

This type attribute is set for each pin on a symbol. It contains a list of pins numbers for each unit in a package. The list is separated by commas, e.g. "1,4,9,12". For instance if the symbol was for a 2-input quad nand gate then one of the input pins would have a VisPin.List attribute of "1,4,9,12". For the "a" unit the pin number is "1"; for the "b" unit the pin number is "4" and so on.

The Permutable Attribute

The permutability of pins on a device will be indicated by an attribute in the library type record.

1. The attribute used is "Permutable".

2. The value of the Permutable attribute is a string of bracketed pin numbers.

3. There are two types of brackets:

   '(' ')' enclosing a list of permutable pins.

   '[' ']' enclosing a list of non permutable pins.

4. Pin lists are separated by ',' characters.

5. Pins that are not in the list are not permutable.

6. The permutable and non-permutable brackets are nested to indicate conditions where individual pins are not permutable but as a group they may be.

   i) A list of pins inside a () are permutable. e.g. (1,2,3).

   ii) A list of pins inside a [] are not permutable. e.g. [1,2,3].

   iii) Lists of non permutable pins [1,2,3] inside a () indicates that the individual non permutable pins may be permuted as a group. e.g. ([1,2,3],[4,5,6])

   iv) Complex permutations as in Example 3 below may indicate that within a non permutable unit some of the pins may be permuted. Such as the A and B inputs on a NAND gate e.g. (1,2) but the output Y must not be swapped with another pin without swapping both A and B. e.g. [(1,2),3] so we get a list like ([(1,2),3],[(4,5),6]).

   This means pins 1 and 2 may be swapped (they are functionally identical). Pins 4 and 5 may be swapped. If we swap pin 3 or 6 we must swap 1,2,3 or 4,5,6 at once.

Example 1

A 54LS74 has two D type flip flop units each identical. To permute one pin we must permute all pins on one FF with the ones on the other FF. This is indicated as follows:

Permutable = "([14,1,2,3,13,12],[8,7,6,5,9,10])";

pins 14,1,2,3,13,12 are non permutable with each other.

pins 8,7,6,5,9,10 are not permutable with each other.

But as a group pins [14,1,2,3,13,12] are permutable with pins [8,7,6,5,9,10].

Example 2

A 54LS04 Hex Inverter. The two pins on each device are not permutable since one is an input and the other is an output. But each of the six inverters can be exchanged as a whole with another.

Permutable = "([1,14],[3,2],[5,6],[7,8],[9,10],[13,12])";

Example 3

A 54LS00 Quad 2-Input Nand Gate. The 4 individual units of the NAND may be permuted as a whole as well the two inputs of each may be permuted. This is indicated as follows:

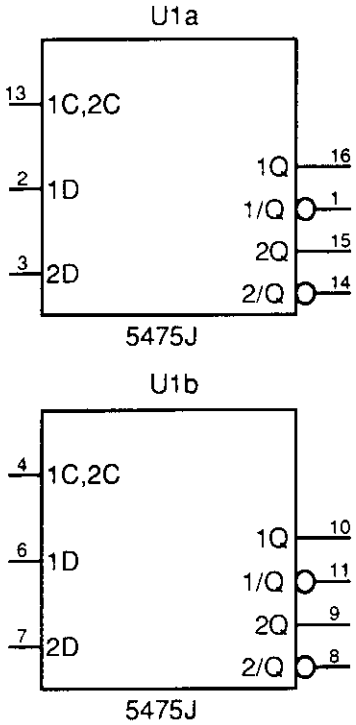
Permutable = "([(1,2),3],[(6,7),5],[(9,10),8],[(12,13),14])";

In this example each non permutable group "[(1,2),3]" is one of the NAND gate units for this unit pins 1 and 2 are permutable. Pin 3 is not. However each unit may be permuted as a whole.

Example 4

A 5475J 4-Bit Bistable Latch. There are 4 nearly identical latches in this package. Each has a D input, a Q and \Q output and an enable (C). However there are only 2 enables for the 4 bits. Bits 1 and 2 have a common enable (Enable 1-2) and bits 3 and 4 have a common enable (Enable 3-4). This differentiates bits 1 and 2 from bits 3 and 4. The D,Q,\Q pins on bit 1 may be exchanged with the same pins on bit 2, but if we exchange bit 1 for bit 3 or 4 we must also exchange the Enables. This is indicated as follows:

```
([13,([2,16,1],[3,15,14])],[4,([6,10,11],[7,9,8])])
```



Package Codes

Package codes have three parts:

Manufacturer Name

Number of Pins

Package Style code

The manufacturer name may be any string of characters although it is advised that no numeric digits should be used since that may confuse the format, the first digits in the package name should be the number of pins used in the package.

Example

TI14DPN

This package is manufactured by Texas Instruments (TI) there are 14 pins on the device and the style is a narrow, plastic dual in line package.

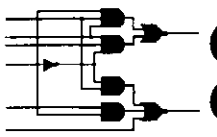
In general the physical dimensions of the packages are not defined by these codes. The DIP package types are standard, others may vary with the manufacturer. The package codes define the sets of foot prints required for PCB layout, it is up to the PCB layout tool to define the actual physical dimensions for the package foot print.

The Package styles used by DesignWorks™ are as follows:

Package Type	**Code**
DIP, plastic narrow	DPN
DIP, plastic medium	DPM
DIP, plastic wide	DPW
DIP, plastic x-wide	DPX
DIP, ceramic narrow	DCN
DIP, ceramic medium	DCM
DIP, ceramic wide	DCW
DIP, ceramic x-wide	DPX
DIP, side brazed narrow	DBN

DIP, side brazed medium	DBM
DIP, side brazed wide	DBW
DIP, side brazed x-wide	DBX
DIP, windowed narrow	DVN
DIP, windowed medium	DVM
DIP, windowed wide	DVW
DIP, windowed x-wide	DVX
MODULE - Leadless/single side	M
MODULE - Leadless/double side	MD
MODULE - Single Sided with Edge Clipped Leads	E
MODULE - Double Sided with Edge Clipped Leads	ED
MODULE - Single Sided with Pin Centered	I
MODULE - Double Sided with Pin Centered	ID
ZIP	Z
Flatpack	F
Flatpack, windowed	FV
Flatpack, Sidebrazed	FB
Quad Flatpack	Q
SOIC narrow	SN
SOIC medium	SM
SOIC wide	SW
SOIC x-wide	SX
Quad SOIC, Gullwing	G
Quad SOIC, Gullwing/Rectangular	GR
SOJ (J-bend SOIC)	J
20/26 pin SOJ	J
SOIC, ceramic	JC
Pin Grid Array	A
Pin Grid Array, windowed	AV
Pin Grid Array, plastic	AP
Pin Grid Array, cavity down	AD
Pin Grid Array, with Heatsink	AH
Leadless chip carrier, ceramic Square	L
Leadless chip carrier, ceramic Rectangular	LR
Leadless chip carrier, sidebrazed Square	LB

Leadless chip carrier, sidebrazed Rectangular	LBR
Leadless chip carrier, windowed Square	LV
Leadless chip carrier, windowed Rectangular	LVR
Leadless chip carrier, plastic Square	P
Leadless chip carrier, plastic Rectangular	PR
Leadless chip carrier, ceramic Square	C
Leadless chip carrier, ceramic Rectangular	CR
Leadless chip carrier, ceramic windowed Square	CV
Leadless chip carrier, ceramic windowed Rectangular	CVR
Leadless chip carrier, plastic with bizarre attributes	PQ



Capilano Computing

DesignNotes

#DW3.0-1, Sept. 21, 1992

Attribute Tips & Techniques for DesignWorks™ 3.0

The purpose of this note is to provide some tips on using the Attributes features of the DesignWorks Schematic tool. In particular, the following topics are covered:

- Using the Duplicate, Merge & Delete functions to make temporary or permanent changes to the display of attributes throughout a design.
- Tips for dealing with attributes when importing designs from older versions or when merging designs from multiple sources.
- How the Schematic tool deals with attribute field definitions when devices are placed from a library or data is copied via the clipboard.

The Duplicate, Merge & Delete Attributes Functions

The Duplicate, Merge and Delete Attribute functions do more than just add or delete a name in the attribute list. They will actually scan the entire design and update the data stored with each object. This makes them very powerful tools for updating your design.

First we will review how the functions are invoked, then mention some possible uses for them.

Using Duplicate, Merge & Delete

These three functions are all invoked from the Define Attribute Fields command at the bottom of the Options menu. No design data is modified until the OK button is clicked in the Define Attribute Fields box.

> NOTES:
>
> 1) When the design is updated, all Duplicates are done first, then all Merges and Deletes. This means that a field can be duplicated, then the original source deleted, all in one invocation of the Define Attributes box. IMPORTANT: There is no guarantee of the order of execution within the Duplicates or within the Deletes and Merges. Therefore, chains of operations (e.g. making a copy of a copy) are not guaranteed to work as expected. If there is any doubt, perform one operation, then exit the Define Attributes box using the OK button, then perform the next operation.
>
> 2) Each of these operations can result in data being updated throughout a design. For large designs there may be a considerable delay after the OK button is clicked in the Define Attribute Fields box.

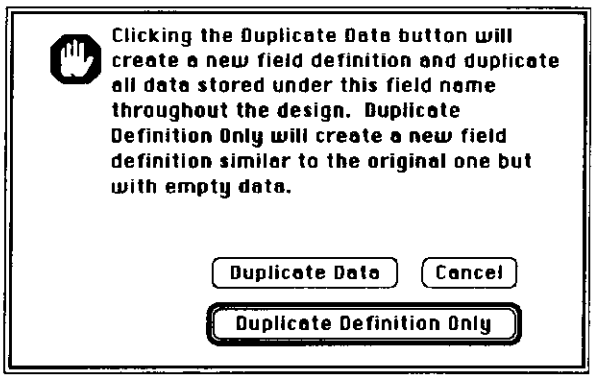
Duplicate

The Duplicate operation adds a new field name to the list with characteristics the same as an existing one. It will also optionally duplicate the data stored under that name throughout the design.

> NOTE: The pre-defined fields in DesignWorks cannot be renamed or removed. However, you can duplicate them, and the copy will be marked as "user-defined", meaning that it can be deleted again or modified in any way.

A field is duplicated by selecting its name in the field list and then clicking on the Duplicate button in the Define Attributes dialog:

At this point a box will be displayed allowing you to choose one of the following options:


Clicking the Duplicate Data button will create a new field definition and duplicate all data stored under this field name throughout the design. Duplicate Definition Only will create a new field definition similar to the original one but with empty data.
Duplicate Data
Cancel
Duplicate Definition Only

Duplicate Definition Only	A new field will be defined with characteristics just like the one being duplicated, but it will have an empty value in each of the objects in the design.
Duplicate Data	A new field will be defined and the associated value in each object in the design will also be duplicated and associated with the new field name. Changing the value of the new field in a given object will have no effect on the original field.

One important feature of this Duplicate operation is that any attribute display options, such as "Visible by Default" or "Show with Field Name", that are turned on before leaving the Define Attributes box will be applied to the duplicated data. This means that a field that was formerly hidden can be duplicated and displayed throughout the entire design with just this one operation.

Merge

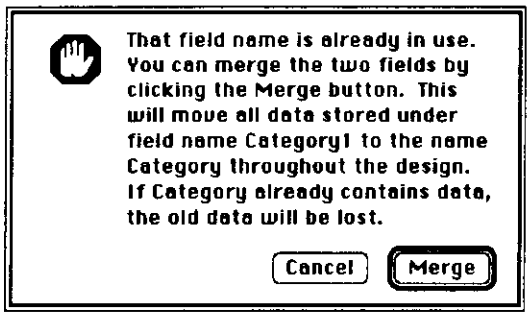
The Merge operation takes two existing attribute field definitions and merges them into one. If a given object (device, signal, pin or design) has

values in both of the original fields, the one being renamed takes precedence and the other is lost.

NOTE: Pre-defined fields cannot be renamed, so they cannot be merged into other fields. You can however merge a user-defined field into a pre-defined one.

There is no explicit button for the Merge operation. Merge is invoked whenever a field is renamed so that its name matches another field's. The procedure is as follows:

- Click on the field to be renamed in the field list. The data in this field will take precedence when the design is updated.
- Type the new name (i.e. the name of the field you wish to merge to) into the Field Name text edit box.
- Type the Tab key or click on any of the option switches to indicate that you have finished typing.
- A box will be displayed asking you to confirm the merge. Click the Merge button:


That field name is already in use. You can merge the two fields by clicking the Merge button. This will move all data stored under field name Category1 to the name Category throughout the design. If Category already contains data, the old data will be lost.
Cancel
Merge

The actual merging of design data does not take place until you click on the OK button in the Define Attributes box.

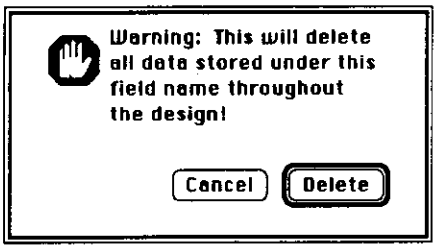
NOTE: You can only merge fields that have similar "Allowed in Object Types" settings. E.g. you cannot merge a signal field into a device field.

Delete

The Delete operation removes the selected name from the list of available fields and removes all data values associated with the field throughout the design. This operation cannot be undone and should be used with care. Pre-defined fields cannot be deleted.

To delete a field, simply select its name in the field list and click on the Delete button:

The following box will be displayed to confirm the Delete operation:



In any case, no design data is updated until the OK button is pressed on the Define Attributes box.

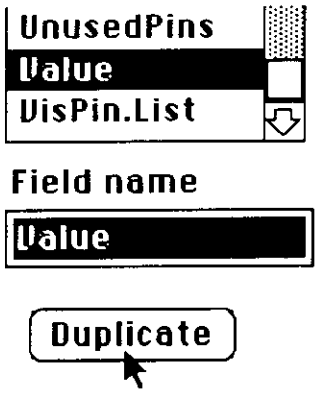
Temporarily Displaying Attributes

The Duplicate function is a convenient method of displaying data temporarily on the schematic. For example, you may wish to display some simulation parameters on the schematic while tracking a particular design problem, even though you don't want them there for the final printout.

To do this:

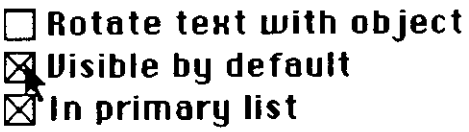
- Select the Define Attribute Fields command in the Options menu.
- Choose the field you wish to display.

• Click on the Duplicate button.



• Click on the Duplicate Data option in the confirmation box.

• Enable the "Visible by Default" and (if desired) the "Show Field Name" and "In Primary List" options.



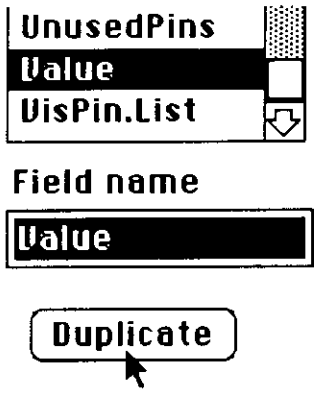
• Click the OK button in the Define Attributes box.

The duplicate field will now be displayed on each object in the design. When the duplicate field is no longer needed, it can be removed with the Delete function described above.

Permanently Showing Data Throughout a Design

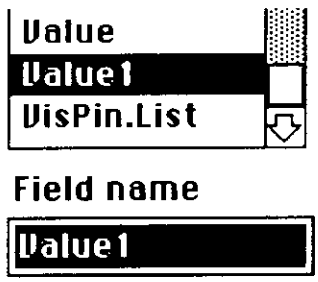
The "Visible By Default" option in the Define Attributes box normally has no effect on values that already exist in the design. Enabling the option for a field that is already in use affects only future entries. However, the Duplicate and Merge operations can be used to update the display of existing data throughout the design. This is done as follows:

- Select the Define Attribute Fields command in the Options menu.
- Select the field to be displayed in the field list.
- Click on the Duplicate button:



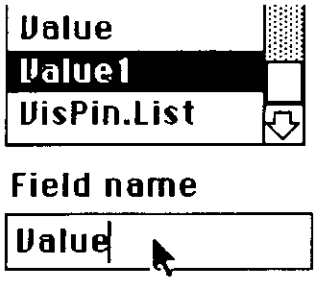
- In the confirmation box, click on the Duplicate Data button:

You will now have a copy of the original field:



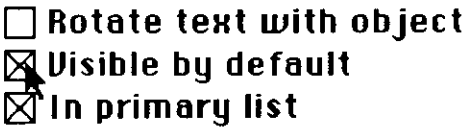
- Make sure the newly-created field is selected in the list, then rename it back to the name of the original field. E.g. If the original field was "Value", the duplicate will be "Value1" and we now select "Value1"

and rename it back to "Value". Press the Tab key when you are finished typing.



• The Merge confirmation box will appear. Click the Merge button.

• Click on the "Visible By Default" option and any other display options desired.



• Click the OK button.

This operation in effect re-enters all the data into the selected field with the new display options.

Merging Dissimilar Designs

The Merge feature is a useful tool when bringing together designs created with different attribute definitions, or when importing designs from other systems or older versions of DesignWorks.

We strongly recommend reviewing attribute field usage under these circumstances and using the Merge feature to convert all non-standard fields to pre-defined ones where possible. This will make the design more compatible with the standard report formats provided with DesignWorks and will reduce confusion in future design revisions.

Importing Attribute Definitions

Whenever circuit date is brought into a design, either from a design file, a device library or the clipboard, it is possible that the incoming data contains attribute fields that are not defined in the design. This section discusses how DesignWorks deals with this situation.

In general, DesignWorks attempts to ensure that an entry exists in the attribute definition table for every field used in the design. When new data is imported, field usage in the imported data is compared to the existing table. Normally, no discrepancies are found and this process is invisible. If any mismatch is found, you will be warned and given a chance to add the new definitions to the table automatically.

Pasting from the Clipboard or Placing a Library Device

In these two cases, DesignWorks does the following:

- All fields used in the incoming device or circuit scrap are looked up in the design's attribute table.
- Any field that doesn't exist in the design is put on a list to be added.
- Any field that does exist in the design but is incompatible (i.e. different "object type" or "keep with instance" settings) is renamed by adding an underscore "_" to the name.
- A list of the new fields that will be added is displayed in a warning box. If any of these fields are unexpected, or can be converted to standard fields, the names should be noted at this point. Changes in field usage can be made using the Merge and Delete operations mentioned elsewhere in this note.

Converting Files from Older Versions

DesignWorks versions prior to 3.0 did not have a standard list of pre-defined fields. For this reason, files converted from older versions should be reviewed to ensure that field names are consistent (i.e. spelling and capitalization variations are eliminated) and that standard names are used wherever possible.

The Converter tool makes a number of field translations automatically for commonly-used fields. See the Converter manual for more information.

Changes in Standard Fields

In rare cases, variations in the list of pre-defined fields will occur between versions of DesignWorks. This will normally only come up if you have used a pre-release or custom modified version of the package. DesignWorks checks for conflicts as it reads a design file into memory. If a field defined in the file is marked as pre-defined, but does not exist in the internal table, or vice versa, you will be warned when the file reading process is completed. The attribute table is automatically updated to resolve the conflict and resaving the file will update it to the new format.